Integrating XML with DB2

New Wiki for DB2 Enhancements via PTF

- Regularly check (or Subscribe) to the DB2 for i Updates Wiki!
  - Contains details on new PTFs that deliver new DB2 capabilities
  - Examples:
    - PROGRAM NAME keyword for controlling SQL Triggers Program Name
    - SQL Query Engine 6.1 support for Logical File on FROM clause
    - CONNECT BY 7.1 support for hierarchical queries
  - Wiki URL:

- The wiki is part of a new IBM i zone in IBM developerWorks
XML Overview

XML

• Extensible Markup Language

• A set of rules for encoding documents electronically

• A textual data format

• Widely used for the representation of arbitrary data structures

• Designed to transport and store data

• http://www.w3.org/standards/xml/

• http://www.w3schools.com/xml/default.asp
What is XML?

XML is "self-describing data"

```xml
<book>
  <authors>
    <author id="47">John Doe</author>
    <author id="58">Peter Pan</author>
  </authors>
  <title>Database systems</title>
  <price>29</price>
  <keywords>
    <keyword>SQL</keyword>
    <keyword>relational</keyword>
  </keywords>
</book>
```

XML: Describes data
HTML: Describes display
XML Attributes & Elements

```xml
<book>
  <authors>
    <author id="47">John Doe</author>
    <author id="58">Peter Pan</author>
  </authors>
  <title>Database systems</title>
  <price>29</price>
  <keywords>
    <keyword>SQL</keyword>
    <keyword>relational</keyword>
  </keywords>
</book>
```

**Design Choices**
- Elements can be repeated (e.g. “keyword”, “author”) attributes cannot.
- Elements can be extended (made deeper), e.g. “author”
- Attributes are shorter, can sometimes be stored/processed more efficiently
- When in doubt, use elements rather than attributes

---

Identifying XML Elements

- **Path**: A sequence of XML tags that identify an XML element or attribute

> Path for ShipDate element:

```
<Order/Part/Shipment/ShipDate
```
Comparison of Relational & XML

Relational
- Externally-Described
- Logical Relationships
- Manual Ordering
- Strong schema
- Strongly typed

XML
- Self-Described
- Hierarchical
- Inherent Ordering
- Schema-variability
- Optionally typed

Phone

<table>
<thead>
<tr>
<th>EMPNO</th>
<th>PHONE</th>
</tr>
</thead>
<tbody>
<tr>
<td>27</td>
<td>406-463-1234</td>
</tr>
<tr>
<td>10</td>
<td>415-010-1234</td>
</tr>
<tr>
<td>10</td>
<td>406-463-4963</td>
</tr>
</tbody>
</table>

Department

<table>
<thead>
<tr>
<th>DEPTID</th>
<th>DEPTNAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>Sales</td>
</tr>
</tbody>
</table>

Employee

<table>
<thead>
<tr>
<th>DEPTID</th>
<th>EMPNO</th>
<th>FIRSTNAME</th>
<th>LASTNAME</th>
<th>PHONE</th>
<th>SALARY</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>27</td>
<td>MICHAEL</td>
<td>THOMPSON</td>
<td>406-463-1234</td>
<td>41250</td>
</tr>
<tr>
<td>15</td>
<td>10</td>
<td>CHRISTINE</td>
<td>SMITH</td>
<td>408-463-4963</td>
<td>52750</td>
</tr>
</tbody>
</table>

Schema Evolution

"Employees are now allowed to have multiple phone numbers…"

Requires:
- Normalization of existing data
- Change of applications

Phone

<table>
<thead>
<tr>
<th>EMPNO</th>
<th>PHONE</th>
</tr>
</thead>
<tbody>
<tr>
<td>27</td>
<td>406-463-1234</td>
</tr>
<tr>
<td>10</td>
<td>415-010-1234</td>
</tr>
<tr>
<td>10</td>
<td>406-463-4963</td>
</tr>
</tbody>
</table>

Department

<table>
<thead>
<tr>
<th>DEPTID</th>
<th>DEPTNAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>Sales</td>
</tr>
</tbody>
</table>

Employee

<table>
<thead>
<tr>
<th>DEPTID</th>
<th>EMPNO</th>
<th>FIRSTNAME</th>
<th>LASTNAME</th>
<th>PHONE</th>
<th>SALARY</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>27</td>
<td>MICHAEL</td>
<td>THOMPSON</td>
<td>406-463-1234</td>
<td>41250</td>
</tr>
<tr>
<td>15</td>
<td>10</td>
<td>CHRISTINE</td>
<td>SMITH</td>
<td>408-463-4963</td>
<td>52750</td>
</tr>
</tbody>
</table>
What is an XML Schema?

- Defines structure, content, data types for XML documents
  - Which elements & attributes are allowed?
  - How do they have be nested, how often can they occur?
  - What are the permissible values for each elements/attribute?
  - Which elements are optional? Which are mandatory?
- Consists of 1 or more schema documents
- Defines a namespace to avoid element name conflicts
- Defines an agreed-upon vocabulary of XML tags for a specific application domain, e.g.
  - HL7 schema in healthcare
  - FpML, FIXML, etc. in finance.
  - etc.

XML Schema example

```xml
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="note">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="to" type="xs:string"/>
        <xs:element name="from" type="xs:string"/>
        <xs:element name="heading" type="xs:string"/>
        <xs:element name="body" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```
XML Integration with DB2 7.1

- Rich XML Support within DB2 for i
- Integrated solution that can replace DB2 XML Extender product
  - New XML data type to simplify storage and retrieval of XML documents
    - XML data access protected with rock-solid DB2 security
    - XML covered by database backup and recovery processes
  - Annotated decomposition of XML documents into DB2 columns
  - Generate XML document with SQL-XML publishing functions
- Advanced, high-speed XML-based search capabilities with IBM OmniFind Text Search Server
XML Data Type

- New XML data type
  - Supports XML values up to 2 GB
  - Type can be used for column, parameter, and host variable values

CREATE TABLE Reservations
  (res_ID INTEGER GENERATED ALWAYS AS IDENTITY,
   res_Doc XML,
   res_TimeStamp TIMESTAMP NOT NULL IMPLICITLY HIDDEN
   FOR EACH ROW ON UPDATE AS ROW CHANGE TIMESTAMP)

XML Data Type

- XML Schema-based validation supported

```sql
INSERT INTO reservations(res_doc)
VALUES(XMLVALIDATE(
XMLPARSE(DOCUMENT
GET_XML_FILE('/dir1/r6.xml'))
ACCORDING TO XMLSCHEMA
ID mylib.resschema))
```

- XML File Reference variables enable simple export XML documents to IFS

```sql
D MY_XMLFILE S SQLTYPE(XML_CLOB_FILE)
/free
     MY_XMLFILE_NAME='/out1.xml';
     MY_XMLFILE_NL = 9;
     MY_XMLFILE_PO = SQFCRT;
     exec sql SELECT res_DOC INTO :MY_XMLFILE
                        FROM reservations WHERE
                        res_ID=1;
     /end-free
```
Integrated XML Utilities

Built-in Functions:

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET_XML_FILE</td>
<td>Returns the contents of an IFS file or source physical file member as a LOB Locator value (requires Commitment Control)</td>
</tr>
<tr>
<td>XMLVALIDATE</td>
<td>Validates XML value against an XML schema</td>
</tr>
<tr>
<td>XMLPARSE</td>
<td>Parses Character/LOB data to produce XML value</td>
</tr>
<tr>
<td>XMLSERIALIZE</td>
<td>Converts XML value into Character/LOB data</td>
</tr>
<tr>
<td>XSLTRANSFORM</td>
<td>Convert XML data into other XML, HTML, and plain text formats using the XSLT processor (requires 5733-XT2)</td>
</tr>
</tbody>
</table>

System Stored Procedures (SYSPROC library):

<table>
<thead>
<tr>
<th>Procedure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XSR_REGISTER</td>
<td>Add an XML Schema document into the DB2 XML Schema Repository (XSR) for Validation / Decomposition</td>
</tr>
<tr>
<td>XSR_ADDSHEMADOC</td>
<td>Merge an XML Schema within an existing XML Schema</td>
</tr>
<tr>
<td>XSR_COMPLETE</td>
<td>Complete the registration of XML Schema(s) within DB2 XSR</td>
</tr>
<tr>
<td>XSR_REMOVE</td>
<td>Remove a registered XML Schema document</td>
</tr>
<tr>
<td>XDBDECOMPXML</td>
<td>Decompose an XML document into specified DB2 objects using annotated decomposition</td>
</tr>
</tbody>
</table>

Storing XML versus “Just Business Data” in DB2

Factors influencing data store choice

- When data design frequently changes ✓ XML
- When you need maximum performance for data retrieval ✓ Data
- When data is processed later as relational data ✓ Data, Data+XML
- When data components have meaning outside a hierarchy ✓ Data
- When data attributes vary, or apply to only a small subset of the rows ✓ XML
- When referential integrity is required ✓ Data
- When the data needs to be updated often ✓ Data
- When the data has audit or archive requirements ✓ XML, XML+Data
Annotated XML Document Decomposition

1) Map the DB2 and XML document relationships

2) Define mapping in XSD file

3) Register and stored XSD mapping within DB2 XML Schema Repository (XSR)

4) Decompose/Shred the XML document

Decomposition Example

```
<shipment>
  <author id="22">
    <name>Tony Dungy</name>
    <book isbn="1414318014">
      <title>Quiet Strength</title>
      <publisher>Tyndale House</publisher>
    </book>
    <book isbn="1414326815">
      <title>Uncommon</title>
      <publisher>Tyndale</publisher>
    </book>
  </author>
</shipment>
```

XSD

```
<xs:element name="shipment">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="author" type="authorType" maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

```
<xs:complexType name="authorType">
  <xs:sequence>
    <xs:element name="name" type="xs:string" db2-xdb:rowSet="AUTHORS" db2-xdb:column="NAME" />
    <xs:element name="book" type="bookType" maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="id" type="xs:integer" db2-xdb:rowSetMapping>
    <db2-xdb:rowSet>AUTHORS</db2-xdb:rowSet>
    <db2-xdb:column>ID</db2-xdb:column>
  </xs:attribute>
</xs:complexType>
```

```
<xs:complexType name="bookType">
  <xs:sequence>
    <xs:element name="title" type="xs:string" db2-xdb:rowSet="BOOKS" db2-xdb:column=BTITLE" />
    <xs:element name="publisher" type="xs:string" />
    <xs:attribute name="isbn" type="xs:integer" db2-xdb:rowSet="BOOKS" db2-xdb:column="ISBN" />
  </xs:sequence>
</xs:complexType>
```
Decomposition Example

XML Decomposition Steps:

1) Create XSD file with DB2 to XML mapping

2) Store and register XSD file within DB2 Schema Repository (XSR)

   CALL SYSPROC.XSR_REGISTER('MYLIB', 'BOOKSCHEM', null,
                             GET_XML_FILE('/dir/authbooks.xsd'), null)
   CALL SYSPROC.XSR_COMPLETE('MYLIB', 'BOOKSCHEM', null, 1)

3) Decompose XML Document

   CALL SYSPROC.XDBDECOMPXML
              ('MYLIB', 'BOOKSCHEM', GET_XML_FILE('/mydir/ship1.xml'), null)

Decomposition Generated Statements:

INSERT INTO authors
       VALUES(22, 'Tony Dungy')

INSERT INTO books
       VALUES(22, 'Quiet Strength', 1414318014),
              (22, 'Uncommon', 1414326815)

SQL XML Publishing Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMLATTRIBUTES</td>
<td>Returns XML sequence that contains an attribute node for each non-null argument</td>
</tr>
<tr>
<td>XMLCOMMENT</td>
<td>Returns XML value with a single comment node from a string</td>
</tr>
<tr>
<td>XMLCONCAT</td>
<td>Returns XML value that represents a forest of XML elements generated by concatenating a variable number of arguments</td>
</tr>
<tr>
<td>XMLDOCUMENT</td>
<td>Returns XML value with a single document node and zero or more nodes as its children</td>
</tr>
<tr>
<td>XMLELEMENT</td>
<td>Returns XML value that represents an XML element</td>
</tr>
<tr>
<td>XMLFOREST</td>
<td>Returns XML value that represents a forest (sequence) of XML elements that all share a specific pattern</td>
</tr>
<tr>
<td>XMLPI</td>
<td>Returns XML value with a single processing instruction node</td>
</tr>
<tr>
<td>XMLNAMESPACES</td>
<td>Returns the declaration of one or more XML namespaces</td>
</tr>
<tr>
<td>XMLROW</td>
<td>Returns XML value with a single document node containing one top-level element node</td>
</tr>
<tr>
<td>XMLTEXT</td>
<td>Returns XML value with single text node that contains value of argument</td>
</tr>
<tr>
<td>XMLAGG</td>
<td>Returns an XML sequence that contains an item for each non-value in set of XML values</td>
</tr>
<tr>
<td>XMLGROUP</td>
<td>Returns XML value with a single document node containing one top-level element node from a group of rows</td>
</tr>
</tbody>
</table>
SQL XML Publishing Example – XMLELEMENT & XMLATTRIBUTE

- Generate XML values for employees celebrating 25th anniversary:

```sql
SELECT XMLSERIALIZE(
    XMLELEMENT(NAME "employee", XMLATTRIBUTES(e.empno as "id"),
                XMLELEMENT(NAME "Name", e.firstnme || ' ' || e.lastname),
                XMLELEMENT (NAME "Extension", e.phoneno),
                XMLELEMENT (NAME "DeptNo", d.deptno)) AS CLOB(100) ) as "XMLResult"
FROM employee e, department d
WHERE e.workdept = d.deptno AND
YEAR(CURRENT DATE) –
    YEAR(hiredate) = 25
```

**Output for XMLResult:**

```
<employee id="000010">
    <Name>JENNA HAAS</Name>
    <Extension>0420</Extension>
    <DeptNo>A00</DeptNo>
</employee>

------------------------------

<employee id="000050">
    <Name>JOSH GEYER</Name>
    <Extension>1103</Extension>
    <DeptNo>E01</DeptNo>
</employee>
```

---

SQL XML Publishing Example - XMLFOREST

- Generate XML values for employees celebrating 25th anniversary using XMLFOREST to simplify query:

```sql
SELECT XMLSERIALIZE(
    XMLELEMENT(NAME "employee", XMLATTRIBUTES(e.empno as "id"),
                XMLFOREST(e.firstnme || ' ' || e.lastname as "Name",
                          e.phoneno as "Extension",
                          d.deptno as "DeptNo")) AS CLOB(100) ) as "XMLResult"
FROM employee e, department d
WHERE e.workdept = d.deptno AND
YEAR(CURRENT DATE) –
    YEAR(hiredate) = 25
```

**Output for XMLResult:**

```
<employee id="000010">
    <Name>JENNA HAAS</Name>
    <Extension>0420</Extension>
    <DeptNo>A00</DeptNo>
</employee>

------------------------------

<employee id="000050">
    <Name>JOSH GEYER</Name>
    <Extension>1103</Extension>
    <DeptNo>E01</DeptNo>
</employee>
```
### SQL XML Publishing Example - XMLAGG

- Generate count and XML value for parts with specified type:

```sql
SELECT COUNT(*) AS PartCnt,
    XMLSERIALIZE(
        XMLELEMENT(NAME "Parts", XMLATTRIBUTES(parttype AS "type"),
            XMLAGG(
                XMLELEMENT(NAME "pid", partid)  ORDER BY partid)
        ) AS CLOB(130)) AS PartList
FROM parts WHERE parttype IN ('C01', 'E21')
GROUP BY parttype
```

<table>
<thead>
<tr>
<th>PartCnt</th>
<th>PartList</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td><code>&lt;Parts type=&quot;C01&quot;&gt;&lt;pid&gt;000130&lt;/pid&gt;&lt;pid&gt;200140&lt;/pid&gt;&lt;/Parts&gt;</code></td>
</tr>
<tr>
<td>3</td>
<td><code>&lt;Parts type=&quot;E21&quot;&gt;&lt;pid&gt;000320&lt;/pid&gt;&lt;pid&gt;100330&lt;/pid&gt;&lt;pid&gt;200340&lt;/pid&gt;&lt;/Parts&gt;</code></td>
</tr>
</tbody>
</table>

### SQL XML Publishing Example - XMLGROUP

- Generate count and XML value for parts with specified type:

```sql
SELECT COUNT(*) AS PartCnt,
    XMLGROUP( parttype AS "type", partid AS "pid"
        ORDER BY parttype, partid
        OPTION ROW "Parts" ROOT "PartList") AS partlist
FROM parts
WHERE parttype IN ('C01', 'E21')
GROUP BY parttype
```

<table>
<thead>
<tr>
<th>PartCnt</th>
<th>PartList</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td><code>&lt;PartList&gt;&lt;Parts&gt;&lt;type&gt;C01&lt;/type&gt;&lt;pid&gt;000130&lt;/pid&gt;&lt;pid&gt;200140&lt;/pid&gt;&lt;/Parts&gt;&lt;/PartList&gt;</code></td>
</tr>
<tr>
<td>3</td>
<td><code>&lt;PartList&gt;&lt;Parts&gt;&lt;type&gt;E21&lt;/type&gt;&lt;pid&gt;000320&lt;/pid&gt;&lt;pid&gt;100330&lt;/pid&gt;&lt;pid&gt;200340&lt;/pid&gt;&lt;/Parts&gt;&lt;/PartList&gt;</code></td>
</tr>
</tbody>
</table>
XML Search

XML Integration with DB2 - Search Capabilities

- **IBM OmniFind Text Search Server**
  - Provides advanced text indexing and high-speed linguistic search capabilities for text values and documents
    - No-charge offering first introduced with IBM i 6.1 release
    - Product number: 5733-OMF
  - XQuery interface not supported, XML search syntax based on XPath standard
  - Enhanced with additional XML search capabilities with IBM i 7.1
  - OmniFind white paper available online at:
    ibm.com/partnerworld/wps/whitepaper/i5os
XML Search Setup – IBM OmniFind Text Search Index Creation

- A text index is created and managed with system stored procedures
  - OmniFind system stored procedures located in SYSPROC schema
  - The SYSTS_CREATE used for text index creation

  CALL SYSPROC.SYSTS_CREATE(
    'myschema',
    'invoice_textix',
    'myschema.invoices(inv_doc)',
    'FORMAT XML,
    UPDATE FREQUENCY D(*) H(0) M(0)')

XML Search Example

- Find all the invoices that reference books that have title element at any level equal to winning

  SELECT inv_number FROM invoices
  WHERE CONTAINS(inv_doc, '@xpath:"//title[contains("winning")]") = 1

  <invoice>
    <book genre="autobiography" pubdate="2008-05-01"
      ISBN="1-414318-02-2" price="14.99">
      <title>Quiet Strength</title>
      <author>
        <first-name>Tony</first-name>
        <last-name>Dungy</last-name>
      </author>
      <publisher>Tyndale</publisher>
    </book>
    <book genre="sports" pubdate="2003-08-01"
      <title>Racing to Win</title>
      <author>
        <first-name>Joe</first-name>
        <last-name>Gibbs</last-name>
      </author>
      <publisher>Doubleday</publisher>
    </book> ...


XML Search Example

- Find invoices where a book has a price attribute that is less than $15.00 or a genre attribute that is equal to autobiography

```
SELECT inv_number FROM invoices
WHERE CONTAINS(inv_doc, 'invoice/book[@price<15.00 or @genre="autobiography"]' ) =1
```

```
<invoice>
  <book genre="autobiography" pubdate="2008-05-01"
        ISBN="1-414318-02-2" price="14.99">
    <title>Quiet Strength</title>
    <author>
      <first-name>Tony</first-name>
      <last-name>Dungy</last-name>
    </author>
    <publisher>Tyndale</publisher>
  </book>
  <book genre="sports" pubdate="2003-08-01"
    <title>Racing to Win</title>
    <author>
      <first-name>Joe</first-name>
      <last-name>Gibbs</last-name>
    </author>
    <publisher>Doubleday</publisher>
  </book>
...</invoice>
```

---

IBM OmniFind – 7.1 Enhancements

- Enhanced XML search support
  - Date and Date Time comparisons:
    `/Book[pubDate > xs:date("2005-04-15")]
  - Numeric comparisons:
    `/Book[Cost <= 59.95]`
  - Namespace specific searches

- Enhanced Save/Restore capabilities
- Graphical text index management
XML Strategy

Integrated DB2 XML versus DB2 XML Extenders

Question:
- If the new DB2 XML support replaces the DB2 XML Extenders program product, why does the DB2 XML Extender product still exist on 7.1?
- Is there a case where DB2 XML Extenders should be used?

Answer:
- For new users to XML with DB2, or casual users of XML Extenders – the new integrated XML support is best choice; it is also the direction IBM i is moving toward
- For customers who use XML Extenders product heavily, the conversion to the integrated XML support can be involved and time consuming
  - DB2 XML Extenders are based on XML DADs while integrated XML support based on XML Schemas (conversion utilities on IBM developerWorks may help)
  - Integrated DB2 XML lacking the XMLTABLE support that is available with XML Extender table functions
- The intent is to have users of DB2 XML Extender migrate to the integrated DB2 XML support over time
Future XML Support

- There are no plans for xQuery at present – if you need it, tell us

- The next possible function to support is XMLTABLE
  - XMLTABLE is an SQL/XML function that evaluates an expression and returns the result as a relational table

Additional Information

- DB2 for i Websites
  - Home Page: ibm.com/systems/i/db2
  - DeveloperWorks Zone: ibm.com/developerworks/db2/products/db2i5OS
  - Porting Zone: ibm.com/partnerworld/i/db2porting

- Newsgroups
  - USENET: comp.sys.ibm.as400.misc, comp.databases.ibm-db2

- Education Resources - Classroom & Online
  - ibm.com/systems/i/db2/gettingstarted.html
  - ibm.com/partnerworld/wps/training/i5os/courses

- OmniFind Documentation
  - Search for "OmniFind PDF" at: http://publib.boulder.ibm.com/infocenter/systems
  - New OmniFind white paper: ibm.com/systems/i/db2/awp.html

- DB2 for i Publications
  - White Papers: ibm.com/partnerworld/wps/whitepaper/i5os
  - Online Manuals: ibm.com/systems/i/db2/books.html
  - DB2 for i Redbooks: ibm.com/systems/i/db2/reiredbooks.html
Are you experiencing performance problems?
Are you using SQL?
Are you getting the most out DB2 for i?

IBM DB2 for i Consulting and Services
- Database modernization
- DB2 WebQuery
- Database design, features and functions
- DB2 SQL performance analysis and tuning
- Data warehousing and Business Intelligence
- DB2 for i education and training

Contact: Mike Cain  mcaih@us.ibm.com
IBM Systems and Technology Group
Rochester, MN USA